Università della Svizzera italiana

Faculty of Informatics

USI Technical Report Series in Informatics

Concept-Based Semantic Annotation, Indexing and Retrieval of Office-Like Document Units

Saša Nešić[1], Mehdi Jazayeri[1], Fabio Crestani[1], Dragan Gaševic[2]

[1] Faculty of Informatics, Università della Svizzera italiana, Switzerland
[2] School of Computing and Information Systems, Athabasca University, Canada

Abstract

We present an ontology-driven approach to semantic annotation, indexing and retrieval of document units. This approach is based on a novel semantic document model (SDM) that we developed to make office-like document units be uniquely identified, semantically annotated with concepts from annotation ontologies and linkable across document boundaries. In the semantic annotation model that we propose, we first lexically expand descriptions of ontological concepts to enhance syntactic matching. Next, we expand a set of syntactic matches with semantically related concepts (i.e., semantic matches) discovered by exploring the annotation ontology. Moreover, we calculate the annotation weight of both the syntactic and semantic matches by taking into account the effects of the lexical expansion and measuring semantic distance between ontological concepts. The retrieval model of document units utilizes the inverted concept index that we generate from the concepts used in the annotation and their weights for document units they annotate. Results of the preliminary evaluation conducted with a prototype implementation are promising. We present the analysis of these results.

Report Info

Published
January 2010

Number
USI-INF-TR-2010-1

Institution
Faculty of Informatics
Università della Svizzera italiana
Lugano, Switzerland

Online Access
www.inf.usi.ch/techreports

1 Introduction

The main goal of ontology-driven information retrieval is to enhance search by making use of available semantic annotations and their underlining ontologies. Accordingly, central to the ontology-driven information retrieval is the problem of having substantial amount of accurate semantic annotations. Most existing semantic annotation approaches [5, 13, 9, 2] are based on syntactic matching of ontological concept labels (descriptions) against document textual content. From the human point of view concept labels often catch the meaning of concepts in an understandable way. However, poor descriptions of ontological concepts, ambiguous meanings of concept labels and inefficiency of existing natural language processing (NLP) techniques usually lead to inaccurate semantic annotation. After the syntactic matching the important part is to discover remaining relevant concepts with the help of formal ontological semantics. Such discovered concepts are usually referred to as semantic matches. The combination of the syntactic and semantic matching can increase the amount of semantic annotations, but it opens the problem of the annotation relevance [16]. Not all semantic matches are equally relevant to the resource they annotate. Therefore, the major issue in this scenario is how to assess the relevance of the discovered semantic matches and to use only the most relevant of them.

Most existing approaches mainly focus on documents as a whole, while we are interested in a finer level of granularity. In this paper we present an approach to semantic annotation, indexing and retrieval of document units defined by a new document model, namely semantic document model (SDM) [10]. We created SDM

to bring the vision of the Semantic Web [1] to office-like desktop documents and to make their units (e.g., paragraphs, sections, tables and figures) to be uniquely identified and linkable across document boundaries. Moreover, SDM document units can be easily put in explicit relations with other digital and non-digital uniquely identified resources. Different logical assertions can be added to these relations as well.

The approach aims to enhance the semantic annotation and indexing of document units by enhancing both syntactic and semantic matching. To enhance syntactic matching we apply *the lexical expansion* of concept descriptions and calculate the weight of each syntactic match. To enhance semantic matching we introduce an algorithm, namely *the concept exploration algorithm*, which explores the annotation ontology starting from the syntactic matches, discovers relevant semantic matches and calculates semantic distances between syntactic and semantic matches. Based on these semantic distances and the weights of the syntactic matches we calculate the weights of the discovered semantic matches. The syntactic and semantic matches together form the concept weight vector of the document unit being annotated. Similarly to document units, we represent the user queries by concept vectors and the corresponding concept weight vectors, and then apply modified vector space model for retrieval of document units.

The organization of the paper is as follows. In Section 2, we discuss related work from ontology-driven information retrieval. In Section 3 we briefly present SDM, focusing only on the aspects which are important to this work. Section 4 describes the semantic annotation in detail. Section 5 discusses the concepts exploration algorithm used in the semantic annotation. In Section 6 we explain our model to indexing and retrieval of the semantically annotated document units. Section 7 provides some details about the prototype development and discusses results of the conducted evaluation.

2 Related Work

In recent years, an increasing number of IR systems have started to use ontologies to come up with semantic representations of documents and to help the users clarify their information needs. The majority of approaches [5, 13, 9, 2, 11] used in these systems is focused on identifying concept instances in documents, based on the concept descriptions (syntactic matches), and then to use the identified concepts for document annotation. Only few approaches try also to assess the relevance/weight of the ontological annotations. In [18] annotation weights are calculated based on the frequency of occurrence of concept instances in the document. The approach presented in [13] does not calculate annotation weights but calculates the weight of each relation instance that associates annotation instances, by analyzing the link structure of the knowledge base. To the best of our knowledge, the issue of determining annotation weights based on ontology features has only been addressed in the approach presented in [14]. This approach extends traditional *tf-idf* method by taking into account the global usage of concepts, individuals and triples in the annotations.

Besides annotation, ontologies have also been used in the process of enriching and disambiguating user queries. Approaches presented in [2, 11] use ontologies as thesauri containing synonyms, hypernyms and hyponyms for the query terms, and do not consider the context of each term, that is, every term is equally weighted. [3] presents a probabilistic query expansion model based on a similarity thesaurus which was constructed automatically. In that approach the query is expanded by adding terms that are similar to the concept of the query, rather than selecting terms that are similar to the query terms. Similarly, in [7], a query expansion is performed by selecting additional terms from those that are connected to the same concepts as the userÕs query terms.

The existing approaches to ontology based IR can be grouped into knowledge based (KB) and vector space (VS) model driven approaches. KB approaches [13, 9] use reasoning mechanisms and ontology query languages to retrieve desired documents from collections of semantically annotated documents. VS approaches [18, 11] try to adapt the traditional IR vector space model by representing both documents and queries by weighted concept vectors. The way in which these vectors are constructed differentiates between the approaches.

To distinguish our work from the related work discussed above we emphasize the main features of our approach as follows. Our approach combines the lexically expanded syntactic matching and ontology features to compose a document concept vector and to calculate concept weights. The concept weights are calculated based on the relevance of the lexically expanded syntactic matches and the semantic distances between concepts in the ontology. The key part of our approach that distinguishes it from similar existing approaches is the concept exploration algorithm, which calculates the semantic distances between concepts in the ontology based on the ontology relationships. Similarly as documents, queries are represented by the

weighted concept vectors and the VS model is applied to calculate similarities between the documents and the queries.

3 Semantic Document Model - SDM

We created a novel semantic document model (SDM), partly inspired by IBM's Darwin Information Architecture (DITA)[1], which divides digital content into small, self-contained topics that can be reused in different deliverables. SDM defines a semantic document as a composite resource built of smaller uniquely identified resources, namely document units (*DUs*) which hold pieces of document content and are semantically annotated by entities (i.e., concepts, properties and instances) from underlined domain ontologies. Moreover, *DUs* can be put in different kinds of relationships with other *DUs* or any uniquely identified digital or non-digital resource. Structural relationships among *DUs* build a document's logical structure. SDM is formally described by the document ontology [10] which defines possible types of *DUs*, types of relationships among *DUs* and provides the annotation interface (e.g., concepts and properties) for adding annotations to *DUs*. Two main types of *DUs* specified by the *do:unitType* property are document content units (*CUs*) and document knowledge units (*KUs*). *CUs* represent units of raw digital content and can be further specialized into *discreteCUs* (e.g., *Graphic* and *TextFragment*) and *continuousCUs* (e.g., *Audio*, *Video* and *Simulation*). *KUs* represent document units that aggregate several *CUs* and add navigation among them (e.g., *Paragraph*, *Section*, *Slide* and *Table*). The annotation interface (Figure 1) consists of the *do:hasAnnotation* property and the *do:DUAnnotation* concept along with its properties. *DU* annotations are instances of the *do:DUAnnotation* concept characterized by the annotation identifier (*do:annotationURI*), the annotation type (*do:annotationType*) property that specifies the type of the ontological entity (i.e., concept, property or instance), the annotation entity (*do:annotationEntity*) property that links the ontological entity to the annotation and the annotation weight (*do:annotationWeight*) that determines the relevance of the ontological entity for the document unit.

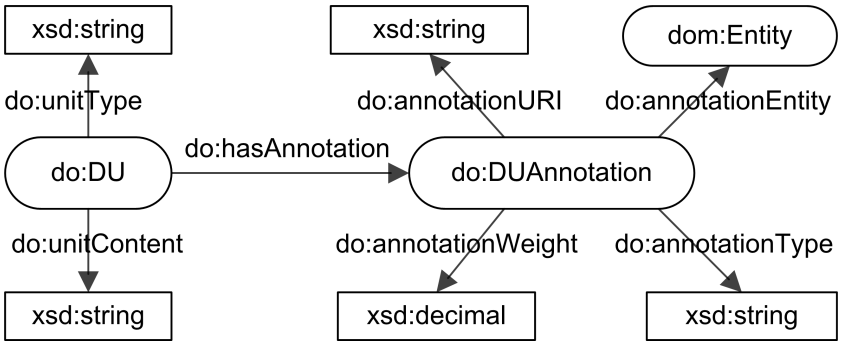

Figure 1: A part of the document ontology (do) that describes the annotation interface for DUs

The semantic document, that is, an instance of SDM is an RDF (Resource Description Framework) graph whose nodes represent instances of *DUs* defined in the document ontology to which the annotations (instances of *do:DUAnnotation*) are linked. Moreover, *DUs* should also hold the units' binary content. However, current implementations of RDF repositories are not meant to store large chunks of binary content, so that we store the binary content of *DUs* into a binary content repository and link it to corresponding RDF nodes via the *do:unitContent* property.

4 Semantic Annotation of Document Units (DUs)

The semantic annotation of *DUs* refers to the process of discovering concepts from domain ontologies whose instances appear in the *DUs* and their linking to *DUs* via the annotation interface. We refer to the domain ontologies whose concepts are used for the annotation as the annotation ontologies in the rest of the paper.

As we pointed out in the Introduction, two major problems with existing semantic annotation approaches are: a) the problem of inaccurate syntactic matching and b) the problem of determining the relevance of both the syntactic and the corresponding semantic matches for *DUs*. In our approach we aim to address

[1] http://www.oasis-open.org/committees/dita

both problems. First, to get more accurate syntactic matches we perform lexical expansion of concept descriptions with lexically related terms from lexical dictionaries. Second, to determine the relevance of the syntactic matches we do the term-weighting of the concept terms regarding the *DU* content and then, based on the term weights, and taking into account the origin of the terms, we calculate the weight of the syntactic matches. To determine the relevance of the semantic matches we calculate the semantic distance between the semantic matches and their corresponding syntactic matches. Then based on the calculated semantic distances and the weight of the syntactic matches, we calculate the weight of the semantic matches.

The semantic annotation is performed in three steps: i) the lexical expansion of concept descriptions, ii) the syntactic matching and iii) the semantic matching. In the rest of the section we describe each step in more detail.

4.1 Lexical Expansion of Concept Descriptions

Any domain ontology can be represented as a graph $O := (\mathbb{C}, \mathbb{R}, H^C, H^R)$ where $\mathbb{C} = \{c_1, c_2, c_3, ..., c_n\}$ is a set of concepts, $\mathbb{R} = \{r_1, r_2, ..., r_m\}$ is a set of relations and H^C, H^R are hierarchies defining a partial order over concepts and relations respectively. Moreover, each concept is described with a set of labels. For example, the set of labels of the concept c_i is $\mathbb{L}_i = \{l_{i1}, l_{i2}, ..., l_{im}\}$. In practice, however, ontology engineers provide only one label for each ontology concept or even neglect to label concepts considering human readable parts of concept URIs to be concept labels [17].

The objective of the lexical expansion is to expand concept descriptions in the annotation ontology with related terms from lexical dictionaries such as WordNet[2]. In our approach we consider three dimensions of lexical relations: synonym, hyponym and hypernym. The results of the lexical expansion are the expanded sets of concept labels. For example, for concept $c_i \in \mathbb{C}$ it is:

$$\mathbb{L}_i^e = \{l_{i1}, l_{i2}, ..., l_{im}, l_{im+1}, ..., l_{ik}\} \tag{1}$$

where the first m labels are original labels from the ontology and the following k are expanded terms discovered by following the synonym, hypernym and hyponym relations respectively. In order to make a distinction between the original labels and those coming from the lexical expansion, we introduce a label relevance factor $rFactor(l)$ and form a concept labels relevance vector. The concept labels relevance vector of the concept c_i is:

$$\overrightarrow{R_L}(c_i) = [rFactor(l_{i1}), ..., rFactor(l_{ik})] \tag{2}$$

where $rFactor(l_{ij}) \in \mathbb{R}$ has value 1 if l_{ij} is the original label, has value δ_{syn} if l_{ij} is the synonym, has value δ_{hyper} if l_{ij} is the hypernym and has value δ_{hypo} if l_{ij} is the hyponym of the original label. In evaluation that we have conducted (Section 7), we used the values $\delta_{syn} = 0.7$, $\delta_{hyper} = 0.47$, $\delta_{hypo} = 0.84$ which had been determined in the experimental studies reported in [6].

4.2 Syntactic Matching

In this step, we analyze the content of *DUs* and check if some of the concept labels, including those from the lexical expansion, appear in *DUs*. For the concepts whose labels appear in *DUs*, (i.e., the syntactic matches) we calculate weights by taking into account the following factors: 1) the concept labels' relevance factor (determined in the lexical expansion), 2) the labels' frequency in the *DU* and 3) the inverse document unit frequency of the concept labels in a collection of all *DUs* annotated by the given annotation ontology. Since SDM (Section 3) defines *DUs* as small pieces of a document content we do not use a length normalization factor in determining the weight of the concept labels. That would be appropriate in the case of the text categorization and indexing of large text documents [15].

Let us consider again the example concept $c_i \in \mathbb{C}$ from Section 4.1 and a document unit d that is being annotated. Firstly, for each label l_{ij} from the expanded set of the concept's labels (1) we count the label frequency $LF(l_{ij})$ in the document unit d. Secondly, we calculate the inverse document frequency $IDF(l_{ij})$ [14] as $log\frac{N}{1+n}$ where n is a number of *DUs* to which the label l_{ij} is assigned and N is a total number of *DUs* in the collection. Finally, when we have $LF(l_{ij})$ and $IDF(l_{ij})$ calculated, we calculate the weight of the concept label l_{ij} for document unit d as follows:

$$w_{l_{ij}} = LF(l_{ij}) * IDF(l_{ij}) \tag{3}$$

[2] http://wordnet.princeton.edu/

The weights of all the concept labels (1) of the concept c_i for the document unit d form a concept labels weight vector:

$$\overrightarrow{W_L}(c_i|d) = [w_{l_{i1}}, w_{l_{i2}}, ..., w_{l_{ik}}] \tag{4}$$

Based on the concept labels weight vector (4) and the concept labels relevance vector (2) we calculate the weight of the concept c_i for the document unit d as a scalar product of these two vectors:

$$w_{c_i} = \overrightarrow{W_L}(c_i|d) * \overrightarrow{R_L}(c_i) \tag{5}$$

If $w_{c_i} > 0$ then the concept c_i annotates the document unit d and w_{c_i} determines the relevance of this annotation.

In the same way as for the concept c_i, we calculate the weights of all the other concepts from the annotation ontology for the document unit d. All concepts with weight greater then zero form the concept vector of the document unit d:

$$\overrightarrow{d} = [c_1, c_2, ..., c_r]; \quad c_i \in \mathbb{C} \quad \wedge \quad w_{c_i} \geq 0 \tag{6}$$

The weights of the concepts from the concept vector $\overrightarrow{d}$ form a concept weight vector of the document unit d:

$$\overrightarrow{W_C}(d) = [w_{c_1}, w_{c_2}, ..., w_{c_r}] \tag{7}$$

As a result of the syntactic matching we get the initial set of the annotation concepts (i.e., syntactic matches) for the *DU* being annotated. These concepts then serve as input concepts for discovering semantic matches.

4.3 Semantic Matching

The objective of the semantic matching is to extend the set of syntactic matches with semantically related concepts from the annotation ontology. For this purpose we introduce the Concept Exploration Algorithm (CEA) explained in detail in Section 5. The algorithm starts from an input concept and traverses the ontology graph to discover semantically related concepts. In short, the algorithm calculates semantic distances between the input concept and the other ontology concepts within a given path distance (i.e., number of hops in the ontology graph) and retrieves those concepts whose semantic distance from the input concept is less than a given semantic distance constraint (i.e., a threshold).

By applying the algorithm to all syntactic matches (6) for the document unit d we discover the set of the document unit's semantic matches and form the expanded concept vector $\overrightarrow{d}^e = [c_1, c_2, ..., c_r, c_{e1}, ..., c_{em}]$ of the document unit. For each of the semantic matches c_{ej} the algorithm calculates the semantic distance $SDist^c(c_{ej}, c_i)$ from the initial syntactic match $c_i \in \overrightarrow{d}$. The weight $w_{c_{ej}}$ of the semantic match c_{ej} for the document unit d is then calculated by the following formula:

$$w_{c_{ej}} = w_{c_i} * \beta^{-SDist^c(c_{ej}, c_i)}; \quad \beta > 1 \tag{8}$$

where w_{c_i} is the weight of the syntactic match c_i and β is a generic coefficient. We devised the formula (6) so that it satisfies boundary conditions regardless of the value of coefficient β. For the first boundary condition $SDist^c(c_{ej}, c_i) = 0$, meaning that the concepts c_{ej} and c_i are semantically identical, $w_{c_{ej}} = w_{c_i}$, that is, the weight of the semantic match is the same as the weight of the initial syntactic match. For the second boundary condition $SDist^c(c_{ej}, c_i) \to \infty$, meaning that the concepts c_{ej} and c_i are semantically unrelated, $w_{c_{ei}} \to 0$, that is, the weight of the semantic match tends towards zero. For $SDist^c(c_{ej}, c_i) \in (0, \infty)$, the optimal value of coefficient β has to be experimentally determined. In the evaluation we report in Section 7, we used the exponential constant e as the value of coefficient β thus (8) belongs to the family of negative exponential functions.

5 Concept Exploration Algorithm - CEA

The main assumption on which the algorithm is based is the possibility to associate numerical values to ontological relations, that we refer to as *the relation semantic distances ($SDist^r$)*, and form the weighted ontology graph. We distinguish between two types of relation semantic distance: $SDist^r_{\mathscr{D} \to \mathscr{R}}(r)$ determining semantic distance of the concepts belonging to the domain ($\mathscr{D}$) of r from the concepts belonging to the range ($\mathscr{R}$) of r and $SDist^r_{\mathscr{R} \to \mathscr{D}}(r)$ determining the semantic distance of the concepts belonging to the range of r from the concepts belonging to the domain of r.

A measuring of the semantic distance/relatedness has received a great deal of attention in the field of lexical semantics [12]. In the field of ontology engineering, however, the focus has been on the formal representation of relations between concepts rather then measuring and quantification of the relational semantic distances. To the best of our knowledge none of the existing ontology representational languages has built-in constructs/attributes that could be used to express the value of the semantic distances between ontological concepts linked by a given relation. In general, the values of the relational semantic distances can be: 1) specified at design time of the ontology by the domain experts, 2) experimentally devised by using a controlled knowledge/data base and 3) learned over time by exploiting the ontology in real world applications within the ontology domain. Based on our experience the choice between these three strategies is strongly domain-dependent. A combination of the strategies is also possible.

The general idea of the algorithm (see Algorithm 1) is to explore the ontology graph starting from the input concept to find all concepts which satisfy the given semantic distance constraint (SD_c) and the given path length constraint (PL_c). SD_c is the maximum allowed semantic distance between the input and target concepts. PL_c is the maximum number of hops (i.e., ontology relations) allowed to belong to a path between the input and target concepts.

Algorithm 1 Concept Exploration Algorithm

1: INPUT O_w, c, SD_c, PL_c
2: OUTPUT $\overrightarrow{C'_e}, \overrightarrow{SD_e}$
3: $\mathbb{P} = Paths1(O_w, c, PL_c) = \{p_1, ..., p_m\}$ {finds all paths from c with a length $\leq PL_c$}
4: $\mathbb{C} = Concepts(\mathbb{P}) = \{c_1, ..., c_n\}$ {extracts all concepts from the set of paths $\mathbb{P}$}
5: **for all** c_i such that $c_i \in \mathbb{C}$ **do**
6: $\quad \mathbb{P}_i = Paths2(c, c_i, \mathbb{P}) = \{p_{i1}, ..., p_{ik}\}$ {finds a set of acyclic paths $\mathbb{P}_i \subset \mathbb{P}$ between c to c_i}
7: $\quad$ **for all** p_{ij} such that $p_{ij} \in \mathbb{P}_i$ **do**
8: $\quad\quad SDist^p(p_{ij})$ {calculates the semantic distance of path p_{ij}}
9: $\quad$ **end for**
10: $\quad SDist^c(c_i, c)$ {calculates the semantic distance of the concept c_i from c}
11: **end for**
12: $\overrightarrow{C'} = [c'_1, ..., c'_p]$, $c'_i \in \mathbb{C}$ and $SDist^c(c'_i, c) \leq SD_c$
13: $\overrightarrow{SD} = [SDist^c(c'_1, c), .., SDist^c(c'_p, c)]$

The algorithm takes the following input: the weighted ontology graph O_w formed by associating values of the relation semantic distances to the ontology relations; the input concept c; the semantic distance constraint SD_c and the path length constraint PL_c. The output consists of a vector of discovered related concepts $\overrightarrow{C'}$ and a vector of the semantic distances $\overrightarrow{SD}$ between the discovered concepts and the input concept. The algorithm starts by the $Paths1(O_w, c, PL_c)$ function (line 3) which constructs a set $\mathbb{P}$ of all possible acyclic paths, starting from the input concept c whose length is $\leq PL_c$. Next, (line 4) the $Concepts(\mathbb{P})$ function extracts all concepts from the set of paths $\mathbb{P}$ and forms a distinct set of extracted concepts $\mathbb{C}$. Next, (line 6) for each concept $c_i \in \mathbb{C}$ function $Paths2(c, c_i, \mathbb{P})$ returns a set of paths $\mathbb{P}_i$ ($\mathbb{P}_i \subseteq \mathbb{P}$) which start in concept c and end in concept c_i. Next, (line 8) for each path $p_{ij} \in \mathbb{P}_i$ between c and c_i, function:

$$SDist^p(p_{ij}) = \sum_{k=1}^{n} SDist^r_{\mathscr{R}\rightarrow\mathscr{D}}(r_k)\Big|_{\text{if direction of } r_k \text{ is } c \rightarrow c_i} \quad \vee \quad SDist^r_{\mathscr{D}\rightarrow\mathscr{R}}(r_k)\Big|_{\text{if direction of } r_k \text{ is } c \leftarrow c_i} \tag{9}$$

calculates the semantic distance of the path that we refer to as *the path semantic distance ($SDist^p$)*. For those $r_k \in p_{ij}$ with the same direction as a direction $c \rightarrow c_i$, function (9) takes $SDist^r_{\mathscr{R}\rightarrow\mathscr{D}}(r_k)$ while for r_k with the direction $c \leftarrow c_i$, it takes $SDist^r_{\mathscr{D}\rightarrow\mathscr{R}}(r_k)$.

After the algorithm calculates the path semantic distances of all paths $\mathbb{P}_i$, it calculates the semantic distance of concept c_i from the input concept c by applying function (10). We call this distance *the concept semantic distance* ($SDist^c$). $SDist^c(c_i, c)$ can be also considered as the relation semantic distance $SDist^r_{\mathscr{R}\rightarrow\mathscr{D}}(r(c, c_i))$ of a new single relation $r(c, c_i)$ from the concept c to c_i.

$$SDist^c(c_i, c) = SDist^r_{\mathscr{R}\rightarrow\mathscr{D}}(r(c, c_i)) = \frac{1}{\sum_{j=1}^{k} \frac{1}{SDist^p(P_{ij})}} \tag{10}$$

We designed function (10) so that it prioritizes the impact of paths with the small path semantic distances in determining the concept semantic distance. Finally, the algorithm discards all concepts from the set $\mathbb{C}$ which do not satisfy the SD_c constraint, forming in that way the output vector of the discovered related concepts $\overrightarrow{C'}$ and the vector of their semantic distances $\overrightarrow{SD}$ from the input concept c.

6 Indexing and Retrieval of Document Units

Based on the concept vectors and the corresponding concept weight vectors generated during the semantic annotation we build an inverted concept index of the semantic documents collection (repository). The index contains a list of concepts (i.e., concept identifiers) from the annotation ontology each of which is assigned a list of document units it annotates. For each document unit in the concept's list, the index also stores the weight of the concept for the document unit.

The search process normally starts with the user constructing a query that reflects his information needs. The initial form of the user query in our approach is a free text query. Constructing free text queries overcomes the problem of knowledge overhead, as it does not require the end user to be familiar with any particular domain ontology. However, the price of free text queries is ambiguity. Keywords may have multiple meanings (lexical ambiguity) and a complex expression can have multiple underlying structures (structural ambiguity). Therefore, the first step in the semantic document search that we propose is *'making sense of the user query'*, that is, finding out the semantic meaning of the query. In our approach we model the semantic meaning of the query by means of a weighted concept vector composed of concepts from the domain ontology. However, it is not easy to find the exact semantic meaning of the query, as there may be more than one concept which matches a single query keyword. Our solution to this is to find out all the concept matches for each keyword and calculate their weights. Actually, the way we form semantic queries from free text queries is quite similar to the semantic annotation of document units. In other words, we treat a free text query the same way as a document unit in the annotation process. After the syntactic matching (Section 4.3) and the semantic matching (Section 4.4), a result is a semantic query represented by a query concept vector and a query concepts weight vector.

Having formed the semantic query, the rest of the search process proceeds as follows. From the concept index we find all document units which match at least one concept from the query concept set. After that, we calculate the similarity between the found document units and the query and rank them. The similarity between the query and the document unit is measured by computing the similarity between the query's concept weight vector and the document unit's concept weight vector, previously reduced to the dimension of the query's concept vector (i.e., the number of concepts in the query's concept vector). Suppose that we have a query q represented by the concept vector $\overrightarrow{q} = [c_{q_1}, ..., c_{q_n}]^N$ and the concept weight vector $\overrightarrow{W_q} = [w_{q_1}, ..., w_{q_n}]^N$, and the document unit d represented with the concept vector $\overrightarrow{d} = [c_1, ..., c_m]^M$ and the concept weight vector $\overrightarrow{W_d} = [w_1, ..., w_m]^M$. The reduced document unit's concept weight vector $\overrightarrow{W'_d} = [w'_1, ..., w'_n]^N$ is formed so that $w'_i = w_j$ if c_i exists in $\overrightarrow{d}$ and $c_i = c_j$. If c_i does not exist in $\overrightarrow{d}$ then $w'_i = 0$. The similarity between vectors $\overrightarrow{W_q}$ and $\overrightarrow{W'_d}$ is computed as the cosine of the angle between them:

$$Similarity(\overrightarrow{W_q}, \overrightarrow{W'_d}) = \frac{\overrightarrow{W}_q * \overrightarrow{W'}_d}{|\overrightarrow{W}_q||\overrightarrow{W'}_d|} \tag{11}$$

The search finishes by ranking the document units based on their similarity to the query and retrieving the ranked list of the document units.

7 Implementation and Preliminary Evaluation

In order to enable the evaluation of our approach we have developed a prototype consisting of two modules: the semantic document authoring module and the semantic document retrieval module. The authoring module transforms MS Office documents (i.e., Word and PowerPoint) into semantic documents (RDF instances of SDM - Section 3) and does the semantic annotation and indexing of *DUs* with a selected domain ontology. For the lexical expansion of concept descriptions the module uses the WordNet.Net - .Net library of the WordNet lexical database. For the text analysis of *DUs* the module uses the Lucene.Net library. The

semantic document authoring starts by the user selecting the annotation ontology that describes the domain of the MS Office document being transformed. The rest of the process is completely automated. The retrieval module takes user defined free text queries, forms the semantic queries and executes them against the indexed collection of semantic documents (i.e., the semantic document repository). The prototype is implemented in C# and it is a part of a broader service oriented architecture, namely the semantic document architecture - SDArch[2], which we have developed previously. SDArch makes the prototype remotely accessible, so that one can author and store semantic documents into remote semantic document repositories as well as search and retrieve *DUs* from the distant repositories. In order to provide a graphical user interface for the prototype we have developed an MS Office add-in called 'SemanticDoc'. The add-in enables MS Office users to transform office documents into semantic documents and to search local and remote semantic document repositories for *DUs* to be potentially reused. Further information, snapshots and demos of the add-in can be found on our project web page[2].

The experimental evaluation that we discuss hereafter, was designed more as a proof of concept; it was not meant to address issues of scalability or efficiency. The document collection that we used in the experiments was composed of 170 Word documents (2735 paragraphs - document units of interest for these experiments) containing records for steel, aluminum, copper, titanium, and other metals. We optioned the collection from KEY-to-METALS[3] company, which maintains one of the world's most comprehensive metals database. As the annotation ontology we used the *Metals* ontology, which we also got from the same company. The ontology contains over 3,500 concepts about metals and their applications. It is an OWL ontology which conforms to the SKOS specification [16]. SKOS defines a family of relations such as $skos:narrower$, $skos:broader$ and $skos:related$ for expressing simple relationships between concepts within an ontology.

Tabel 1 shows a subset of semantic relations in the *Metals* ontology, along with their SKOS and OWL representations and values of the relation semantic distances. The values of the relation semantic distances were assessed based on the results of the experimental studies [6]. In these studies the authors measured the semantic similarity/relatedness between terms in WordNet, connected via the *hypernymy*, *hyponymy*, *holonymy*, *meronymy* and *synonymy* relations, and produced the following values: $\delta_{hyper} = 0.47$, $\delta_{hypo} = 0.84$, $\delta_{holo} = 0.12$, $\delta_{mero} = 0.16$ and $\delta_{syn} = 0.70$. Value $\delta_r = 0$ means that two terms are semantically unrelated via relation r, and $\delta_r = 1$ that the terms are semantically identical. We calculate the values of the relation semantic distances as $1-\delta_r$ and take into account the fact that *hypernymy* and *hyponymy* as well as *holonymy* and *meronymy* are mutually inverse relations. Moreover, the *Metals* ontology contains the $owl:sameAs$ relation which links two semantically identical concepts/individuals, so that both of the relation semantic distances have been assessed as zero.

Semantic relation	Representation	$SDist^r_{\mathscr{R}\to\mathscr{D}}(r)$	$SDist^r_{\mathscr{D}\to\mathscr{R}}(r)$
hypernym	$skos:broader$	$1-\delta_{hyper} = 0,53$	$1-\delta_{hypo} = 0,16$
hyponym	$skos:narrower$	$1-\delta_{hypo} = 0,16$	$1-\delta_{hyper} = 0,53$
holonym	$skos:relatedPartOf$	$1-\delta_{holo} = 0,88$	$1-\delta_{mero} = 0,84$
meronym	$skos:relatedHasPart$	$1-\delta_{mero} = 0,84$	$1-\delta_{holo} = 0,88$
synonym	$owl:equivalentClass$	$1-\delta_{syn} = 0,30$	$1-\delta_{syn} = 0,30$
identical	$owl:sameAs$	0	0
generic sem. relation	$skos:related$	-	-

Table 1: Relation semantic distances in Metals ontology

In order to evaluate our approach, we have transformed the evaluation document set into semantic documents with five different annotation/indexing options:

- AO_1 - simple syntactic matching,
- AO_2 - lexically expanded syntactic matching,
- AO_3 - lexically expanded syntactic matching and semantic matching ($SD_C = 1$),
- AO_4 - lexically expanded syntactic matching and semantic matching ($SD_C = 2$),
- AO_5 - lexically expanded syntactic matching and semantic matching ($SD_C = 3$).

The first option AO_1 - simple syntactic matching is present in most of the existing ontology-driven information retrieval approaches [19, 8, 4, 20]. The second option AO_2 includes again only syntactic matching but

[2] http://www.semanticdoc.org
[3] http://www.keytometals.com/

now with the lexically expanded concept descriptions. The last three options AO_3, AO_4 and AO_5 comprise all the features (i.e., lexical expansion, syntactic matching and semantic matching) of the proposed semantic annotation and indexing. They only differ in the value of the SD_C (semantic distance constraint) parameter of the concept exploration algorithm (Section 5). The value of the path length constraint is fixed at $PL_c = 3$ for these evaluation tests.

As a result of the transformation we obtained five semantic document collections with the corresponding inverted concept indexes. Table 2 shows for each of the annotation options: 1) the number of concepts from the annotation ontology (Metals) that have been used in the annotation and indexing, 2) the total number of syntactic and semantic matches and 3) the average weights of the syntactic and semantic matches based on 20 randomly chosen document units (i.e., paragraphs). First, comparing AO_1 and AO_2 which both implement only syntactic matching, we can see that the lexical expansion of concept descriptions increases the number of concepts involved in the annotation from 211 to 343 and the number of syntactic matches from 1524 to 3182 but also the average weight of syntactic matches from 2.56 to 3.62. In other words these increases show that the lexical expansion improves both the quantity and quality of the annotation. The next three options $AO_3 - AO_5$ produce the same number of syntactic matches as AO_2 (3182), since the syntactic matching stays intact, but they increase the quantity of the annotation by adding certain numbers of semantic matches. While the number of the semantic matches increases from AO_3 to AO_5 (i.e., 6714; 11102; 23716) the average weight of semantic matches decreases (i.e., 2.43; 1.12; 0.27). This was expected as with higher values of the semantic distance constraint (SD_c) we get more but less relevant semantic matches.

Annotation options	Number of concepts	Number of syn. matches	Number of sem. matches	Average weight of syn. matches	Average weight of sem. matches
AO_1	211	1524	-	2.56	-
AO_2	343	3182	-	3.62	-
AO_3	672	3182	6714	3.62	2.43
AO_4	795	3182	11102	3.62	1.12
AO_5	924	3182	23716	3.62	0.27

Table 2: Annotation data for each annotation option (AO_1-AO_5)

To evaluate the performance of the proposed concept-based information retrieval we formed five queries related to the topic of the evaluation document set and asked three of our colleagues from the university to read the documents and mark document units (i.e., paragraphs) relevant for the queries. The queries were then executed against each of the five semantic document collections. Figure 2 shows interpolated precision at standard recall points.

Comparing the P-R curves of AO_1 and AO_2 we can see that the lexically expanded syntactic matching outperforms from the simple syntactic matching in both recall and precision. Moreover, all three options (i.e, AO_3, AO_4, AO_5) which include semantic matching further increase overall precision and recall. Comparing their P-R curves and by knowing that they differ only in the value of the semantic distance constraint (i.e., $SD_c = 1$, $SD_c = 2$, $SD_c = 3$) we can observe that there is an optimal value for the concept semantic distance ($SDist^c$) with regard to optimal precision and recall. It means that the semantic matches with the concept semantic distance higher than the optimal value reduce performances. In our evaluation the optimal concept semantic distance falls in a range between 2 and 3, since the precision of AO_4 is higher than of AO_3 but then it drops for AO_5. However, we believe that the value of the optimal concept semantic distance is strongly dependent on the used evaluation document set and the annotation ontology.

The evaluation results indicate that our concept-based semantic annotation, indexing and retrieval approach: 1) enlarges the amount of semantic annotations and 2) improves the performances of DU_s retrieval not just in terms of recall, as it was to be expected, but also in terms of precision. We plan to perform more large-scale evaluation on document sets from different domains by using different annotation ontologies.

8 Conclusion

In this paper we present an ontology-driven approach to semantic annotation and indexing of office-like document units, which we developed in order to improve the retrieval of such document units. In our approach we form the annotations by combining syntactic matches of lexically expanded ontological concepts with

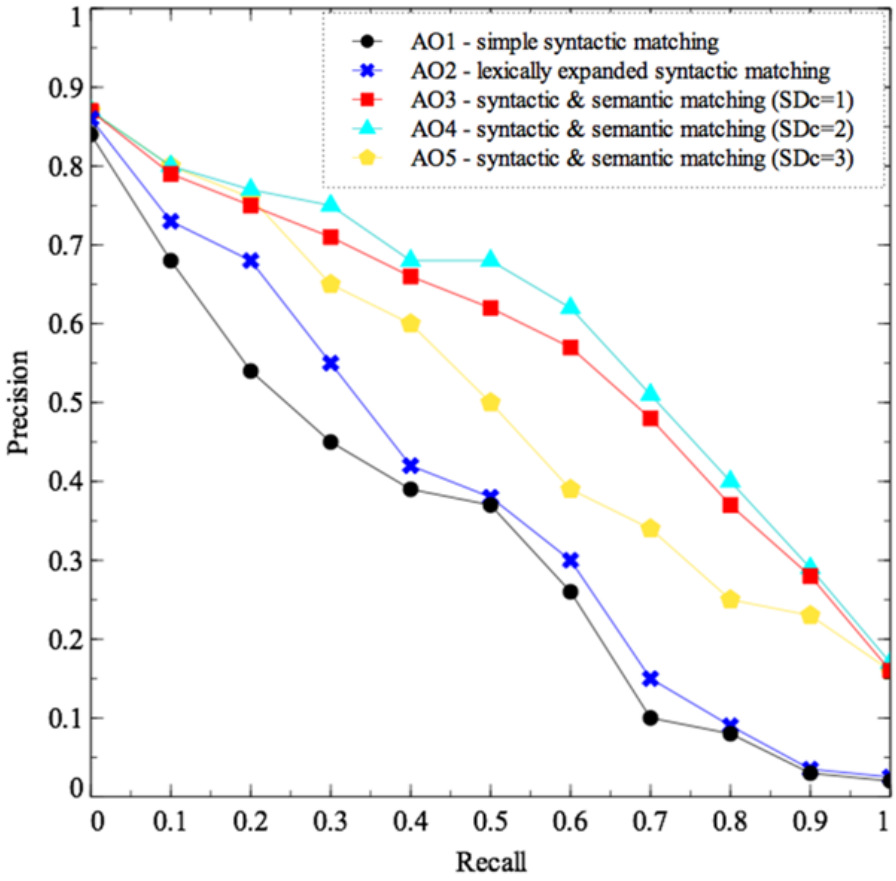

Figure 2: Interpolated precision of AO_1-AO_5 at standard recall points

semantic matches obtained by exploring the ontology graph. For each, either syntactic or semantic match, we calculate its relevance/weight for the document unit it annotates. The annotation weights are used in the indexing of document units and the calculating document units' similarity with the user queries. In order to evaluate the approach we have developed the prototype and conducted a preliminary evaluation. Evaluation results on the chosen document-set and the annotation ontology have shown the improvements of retrieval performance compared to simple syntactic matching which is applied in most existing ontology-driven information retrieval approaches. Our future work will be mainly focused on further evaluation of the proposed approach and its applicability to documents and ontologies from different domains.

References

[1] Tim Berners-Lee, James Hendler, and Ora Lassila. The Semantic Web. *Scientific America*, 2001.

[2] Regina M. M. Braga, Cláudia Maria Lima Werner, and Marta Mattoso. Using Ontologies for Domain Information Retrieval. In *DEXA Workshop*, pages 836–840, 2000.

[3] Yonggang Qiu Department, Yonggang Qiu, and H. P. Frei. Concept Based Query Expansion. In *16th annual int. ACM SIGIR conference on Research and Development in information Retrieval*, pages 160–169, 1993.

[4] Li Ding, Tim Finin, Anupam Joshi, Rong Pan, Scott Cost, Yun Peng, Pavan Reddivari, Vishal Doshi, and Joel Sachs. Swoogle: a search and metadata engine for the semantic web. In *the 13th ACM CIKM*, pages 652–659, 2004.

[5] Jun feng Song, Wei Ming Zhang, Weidong Xiao, Guo hui Li, and Zhen ning Xu. Ontology-Based Information Retrieval Model for the Semantic Web. In *EEE '05: Proceedings of the 2005 IEEE International Conference on e-Technology, e-Commerce and e-Service*, pages 152–155, 2005.

[6] Zhiguo Gong, Chan Wa Cheang, , and Leong Hou U. Multi-term Web Query Expansion Using WordNet. In *the 17th Database and Expert Systems Applications Conference, DEXA*, pages 379–388, 2006.

[7] F. A. Grootjen and Theo P. van der Weide. Conceptual query expansion. *Data Knowl. Eng.*, 56(2):174–193, 2006.

[8] Glen Jeh and Jennifer Widom. Simrank: A Measure of Structural-Context Similarity. In *the 8th Int. Conf. on Knowledge Discovery and Data Mining*, pages 538–543, 2002.

[9] Atanas Kiryakov, Borislav Popov, Ivan Terziev, Dimitar Manov, and Damyan Ognyanoff. Semantic annotation, indexing, and retrieval. *J. Web Sem.*, 2(1):49–79, 2004.

[10] Saša Nešić. Semantic Document Model to Enhance Data and Knowledge Interoperability. *Annals of Information Systems, Springer*, 6:135–162, 2009.

[11] Rifat Ozcan and Y. Alp Aslandogan. Concept-Based Information Access. In *ITCC '05: Proceedings of the International Conference on Information Technology: Coding and Computing*, pages 794–799, 2005.

[12] Philip Resnik. Semantic similarity in a taxonomy: An information-based measure and its applications to problems of ambiguity in natural language. *Journal of Artificial Intelligence Research*, 11(95–130), 1999.

[13] Cristiano Rocha, Daniel Schwabe, and Marcus Poggi de Aragão. A hybrid approach for searching in the semantic web. In *WWW '04: Proceedings of the 13th international conference on World Wide Web*, pages 374–383, 2004.

[14] Tuukka Ruotsalo and Eero Hyvönen. A Method for Determining Ontology-Based Semantic Relevance. In *the 18th Database and Expert Systems Applications Conference, DEXA*, pages 680–688, 2007.

[15] Gerard Salton and Chris Buckley. Term-Weighting Approaches in Automatic Text Retrieval. *Inf. Process. Manage.*, 24(5):513–523, 1988.

[16] Jouni Tuominen, Matias Frosterus, and Eero Hyvönen. ONKI SKOS Server for Publishing and Utilizing SKOS Vocabularies and Ontologies as Services. In *ESWC*, pages 768–780, 2009.

[17] Victoria Uren, Philipp Cimiano, Jose Iria, Siegfried Handschuh, Maria Vargas-Vera, Enrico Motta, and Fabio Ciravegna. Semantic Annotation for Knowledge Management: Requirements and a Survey of the State of the Art. *Journal of Web Semantics*, 4(1):14–28, 2006.

[18] David Vallet, Miriam Fernández, and Pablo Castells. An Ontology-Based Information Retrieval Model. In *ESWC*, pages 455–470, 2005.

[19] Wensi Xi, Edward Fox, and Weiguo Fan. Simfusion: measuring similarity using unified relationship matrix. In *the 28th ACM SIGIR Int. Conf. on Research and Development in Information Retrieval*, pages 130–137, 2005.

[20] Hai Zhuge and Liping Zheng. Ranking Semantic-Linked Network. In *the 12th WWW*, pages 32–34, 2003.

USI Technical Report Series in Informatics

2006

1. Vaide Zuikeviciute, Fernando Pedone
   Conflict-Aware Load-Balancing Techniques for Database Replication
2. Nicolas Schiper, Rodrigo Schmidt, Fernando Pedone
   Optimistic Algorithms for Partial Database Replication
3. Anna Egorova-Frster, Amy L. Murphy
   A Feedback-Enhanced Learning Approach for Routing in WSN
4. Cyrus P. Hall, Antonio Carzaniga, Alexander L. Wolf
   DV/DRP: A Content-Based Networking Protocol for Sensor Networks
5. Antonio Carzaniga, Aubrey J. Rembert, Alexander L. Wolf
   Understanding Content-Based Routing Schemes
6. Jeff Rose, Cyrus Hall, Antonio Carzaniga
   Spinneret: A Log Random Substrate for P2P Networks
7. Paolo Bonzini, Laura Pozzi
   Polynomial-Time Subgraph Enumeration for Automated Instruction Set Extension
8. Lasaro Camargos, Marcin Wieloch, Fernando Pedone, Edmundo Madeira
   A Highly Available Log Service for Distributed Transaction Termination

2007

1. Lasaro Camargos, Fernando Pedone, Marcin Wieloch
   High-Performance Transaction Processing in Sprint
2. Lasaro Camargos, Rodrigo Schmidt, Fernando Pedone
   Multicoordinated Paxos
3. Giovanni Denaro, Alessandra Gorla, Mauro Pezz
   An Empirical Evaluation of Data Flow Testing of Java Classes
4. Nicolas Schiper, Fernando Pedone
   Optimal Atomic Broadcast and Multicast Algorithms for Wide Area Networks
6. Romain Robbes, Michele Lanza
   Towards Change-aware Development Tools
7. Aliaksei Tsitovich
   Understanding Vulnerabilities

2008

1. Nicolas Schiper, Sam Toueg
   A Robust and Lightweight Stable Leader Election Service for Dynamic Systems
2. Nicolas Schiper, Fernando Pedone
   Solving Atomic Multicast when Groups Crash
3. Vaide Zuikeviciute, Fernando Pedone
   Correctness Criteria for Database Replication: Theoretical and Practical Aspects
4. Jochen Wuttke
   Property Templates and Assertions Supporting Runtime Failure Detection
5. Dmitrijs Zaparanuks, Milan Jovic, Matthias Hauswirth
   Accuracy of Performance Counter Measurements
6. Romain Robbes, Michele Lanza, Damien Pollet
   A Benchmark for Change Prediction
7. Paolo Bonzini, Laura Pozzi
   On the Complexity of Enumeration and Scheduling for Extensible Embedded Processors

2009

1. Nicolas Schiper, Pierre Sutra, Fernando Pedone
   Genuine versus Non-Genuine Atomic Multicast Protocols
2. Cyrus Hall, Antonio Carzaniga
   Doubly Stochastic Converge: Uniform Sampling for Directed P2P Networks
3. Nicolas Schiper, Fernando Pedone
   Fast, Flexible, and Highly Resilient Genuine Fifo and Causal Multicast Algorithms
4. Anna Frster, Amy L. Murphy
   FROMS: A Failure Tolerant and Mobility Enabled Multicast Routing Paradigm with Reinforcement Learning for WSNs
5. Jochen Wuttke
   Defining Model Transformations for Property Templates
6. Antonio Carzaniga, Cyrus Hall, Giovanni Toffetti Carughi, Alexander L. Wolf
   Practical High-Throughput Content-Based Routing Using Unicast State and Probabilistic Encodings
7. Domenico Bianculli, Walter Binder, Mauro Luigi Drago
   Automated Performance Assessment for Service-Oriented Middleware